

## Crystal Touch: Single Touch

### User's Manual

#### Table of Contents

1.	Introduction.....	2
2.	Hardware Interface.....	3
3.	Serial Peripheral Interface (SPI).....	4
4.	Touch Detection.....	8
5.	Register Descriptions .....	10
6.	Initialization.....	13
7.	Coordinate Scaling.....	14
8.	Data Ready .....	14
	Appendix A – Software Driver Sample.....	15

Rev. 11 — March 4, 2014

Product descriptions and specifications are subject to change without notice.

## 1. Introduction

This document describes the hardware and software interfaces for Ocular's Crystal Touch: Single Touch line of projected capacitive touch panels. It applies to all Crystal Touch: Single Touch panel sizes.

### 1.1. Theory of Operation

Ocular's Crystal Touch: Single Touch panels are based on a projected capacitive technology; a capacitive field is projected through the top layer of glass. This projection mechanism allows the conductive elements to be embedded inside the glass, protecting them from damage. The top surface of Ocular's Crystal Touch: Single Touch panels is a high quality piece of custom glass designed for optimum clarity and hardness.

When the user places a finger on the glass surface, they alter the capacitive field being projected through the glass. This alteration is detected and used to determine the location of the user's finger.

### 1.2. Design Considerations

The position of the user's finger is determined by calculating the centroid of the changes in the electrical field. With some enclosure designs it may be difficult for the user to effectively move the center of their finger into the corners of the touch panel. For this reason, applications should not place small touch areas in the corners of the screen.

When designing an application to work with a finger-actuated touch panel, it is recommended that the application avoid creating small touch areas around the outer edges of the display. Studies show that for the best possible user experience, touch areas should be at least 10 mm<sup>2</sup> and should be located at least 5 mm from the edge of the screen.

### 1.3. Calibration

Each Crystal Touch: Single Touch panel undergoes an internal calibration when first powered up. Calibration takes around 100 ms.

**The operator must not touch the panel during calibration as this could affect measurement sensitivity.**

Once the touch panel is powered up, there is no need to execute any subsequent calibration cycles unless the operating parameters of the panel are changed (see "4.4. Capacitor Charge Time" for an example of when this is necessary).

## 2. Hardware Interface

### 2.1. Signal Descriptions (SPI)

Ocular Crystal Touch: Single Touch panels with SPI interface have the following hardware pinout:

Table 1

Pin No.	Symbol	Description
1	SCK	SPI Clock
2	MISO	SPI Master In Slave Out
3	SS_N	SPI Select
4	DR	Data Ready
5	MOSI	SPI Master Out Slave In
6	GND	Ground
7	V <sub>DD</sub>	Power Supply
8	GND	Ground
9	SNSN	Stylus input
10	GND	Ground

The mating connector part is Molex 52207-1060 or equivalent.

The signals are described below:

**SCK, MISO, SS\_N, MOSI:** These pins are part of the SPI communications interface. See section 2.3 for details.

**SNSN:** This pin is for a dedicated stylus input. Leave this pin unconnected (contact Ocular for details on using Crystal Touch: Single Touch with a stylus).

### 2.2. I<sup>2</sup>C Interface

An I<sup>2</sup>C interface is available as an alternative to the standard SPI interface. For information on the I<sup>2</sup>C interface refer to the Crystal Touch: Single Touch I<sup>2</sup>C Addendum.

### 2.3. VDD

It is important that the power supply provided to the touch panel be relatively clean, particularly in the low frequency range of 1 kHz to 500 kHz. Noise on the power supply or GND lines can affect the accuracy of the touch panel.

### 2.4. Case Ground

If the system enclosure is metal, the touch panel GND must be connected to the case ground.

### 3. Serial Peripheral Interface (SPI)

#### 3.1. Basic Operation

SPI communication is a four wire bus that includes a slave select line for communicating with multiple devices on the SPI bus. The Crystal Touch: Single Touch controller is designed as a slave device with a select line, two data lines, and a clock line (see Figure 1). The SPI clock signal (SCK) must be initialized to low when idle. To activate the controller's SPI bus, pull the slave select line low (SS\_N). Data sent by the master is placed on the Master Out Slave In line (MOSI). Data returned by the controller is placed on the Master In Slave Out line (MISO). Both sets of data are latched on the falling edge of SCK. Data is presented Most Significant Byte first (MSB). The controller's SPI bus supports data rates up to 4 Mhz.

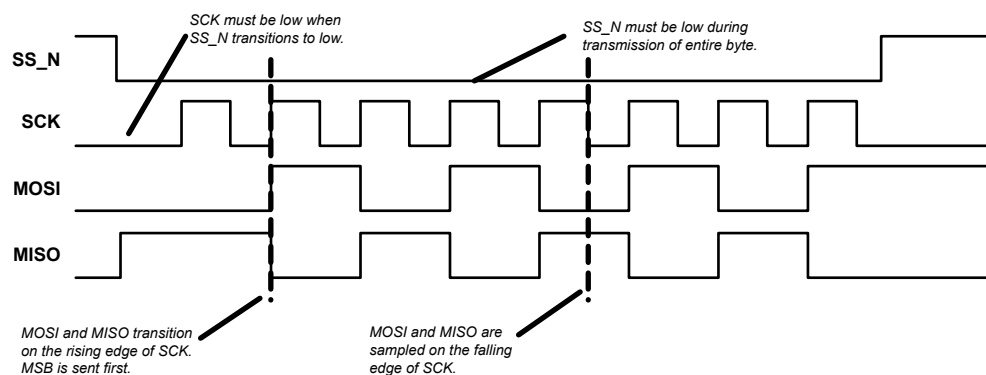


Figure 1: SPI Signals

#### 3.2. SPI Timing Requirements

Timing for the SPI data lines is shown below (see Figure 2). The data exchange is started when the master pulls the SS\_N line low. Due to the full duplex nature of the SPI bus, the Crystal Touch: Single Touch controller always returns data at the same time that it is receiving data. The returned byte may be data from the previous command or it may be a filler byte.

Table 2

Item	Symbol	Min.	Max.	Unit
Frequency	—	0.42	4.0	MHz
Period	TQCYC	0.25	2.38	µs
Clock High or Low Time	T <sub>SW</sub>	TQCYC/2	—	—
Chip Select Lag Time	T <sub>LAG</sub>	0.030	—	µs
Inter-Message Transfer Delay	T <sub>TD</sub>	50	—	µs
Chip Select Lead Time	T <sub>LEAD</sub>	TQCYC	—	—
Slave Data Valid Time	T <sub>V</sub>	—	0.500	µs
Slave Data Hold Time	T <sub>HO</sub>	TQCYC/2	—	—
Master Data Setup Time	T <sub>SU</sub>	TQCYC/4	—	—
Master Data Hold Time	T <sub>HI</sub>	TQCYC/4	—	—

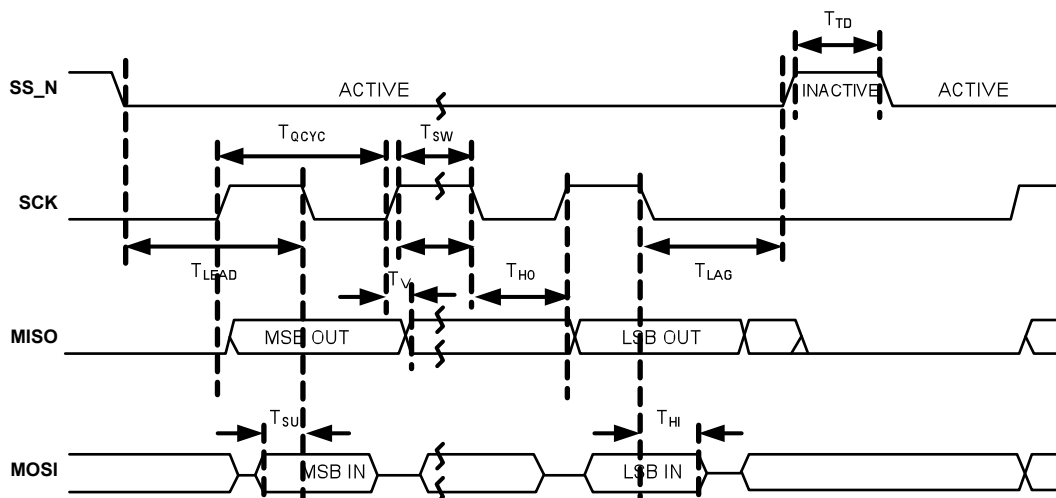


Figure 2: SPI Timing

### 3.3. SPI Inter-Message Transfer Delay

The Crystal Touch: Single Touch controller requires 50  $\mu$ s minimum between transmitted bytes regardless of the baud rate (see  $T_{DI}$  in Figure 2). Some processors include a settable delay time between bytes. For those processors that don't have a settable inter-byte delay time, or for processors whose maximum inter-byte delay time is less than 50  $\mu$ s, the Crystal Touch: Single Touch software driver will have to enforce the inter-byte delay.

### 3.4. SPI Slave Select Line

The slave select line must be low for a least one cycle before the first clock of each byte is transferred. There is no requirement for the slave select line to toggle between bytes or between packets. For example, the slave select line can be driven low at power up and stay low forever. It can also toggle between each byte, as long as it is low for at least one clock cycle before the first clock of each byte.

### 3.5. Register Access Protocol Using SPI

The Crystal Touch: Single Touch controller uses a communication method known as Register Access Protocol (RAP). RAP only has two commands: READ and WRITE. The registers have five bit addresses ranging from 0x00 to 0x1F. Commands are formatted as follows.

Table 3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Read (1) Write (0)	Register Address Bit 4	Register Address Bit 3	Register Address Bit 2	Register Address Bit 1	Register Address Bit 0

### 3.6. Register Addresses

Only nine of the thirty two registers are used. The rest are reserved and should not be read or written. The available registers are:

Table 4

Register Address	Description
0x02	Status
0x03	Power and Reset
0x04	Control
0x07	Calibrate
0x0A	Z Idle
0x14	X Position Low Byte
0x15	Y Position Low Byte
0x16	X and Y Position High Nibbles
0x17	Z Level

### 3.7. Reading From a Register

Reading a register requires four SPI exchanges. The master starts by sending the READ command byte. The simultaneous response byte may be data from the previous command or it may be a filler byte. Three additional bytes need to be sent after the READ byte to give the controller time to process and respond to the command. The response will be sent during the transmission of the final byte. When sending one command at a time, the three additional bytes must be 0xFB (0xFB stands for "filler byte").

**Example:** Read the contents of the Status register (0x02):

*A Read command is formatted as 101xxxxx (0xAX). When OR'ed with the register address, the value is 0xA2:*

$$0xA0 \mid 0x02 = 0xA2$$

*Send 0xA2 followed by 0xFB three times. The response to the third 0xFB byte will be the contents of register 0x02.*

Table 5

Byte	Command (MOSI)	Response (MISO)
1	0xA2	Note 1
2	0xFB	0xFB
3	0xFB	0xFB
4	0xFB	Contents of register 0x02

Note 1: The first response byte may be a filler byte or it may be the response to a previous command.

### 3.8. Pipelining Multiple Reads

Multiple reads can be pipelined for greater throughput by using the auto increment command (0xFC) in place of the filler byte (0xFB)

**Example:** Read the contents of the Position registers (0x14 through 0x16):

*A Read is formatted as 101xxxxx (0xAX). When OR'ed with the first register address, the value is 0xB4*

$$0xA0 \mid 0x14 = 0xB4$$

*Send 0xB4 followed by 0xFC four times. Then send a single 0xFB to terminate the sequence. The response to the third 0xFC will be the contents of register 0x14. The response to the fourth 0xFC will be the contents of register 0x15. The response to the last byte, 0xFB, will be the contents of register 0x16.*

Table 6

Byte	Command (MOSI)	Response (MISO)
1	0xB4	Note 1
2	0xFC	0xFB
3	0xFC	0xFB
4	0xFC	Contents of register 0x14
5	0xFC	Contents of register 0x15
6	0xFB	Contents of register 0x16

Note 1: The first response byte may be a filler byte or it may be the response to a previous command.

### 3.9. Writing To a Register

Writing to a register requires two SPI exchanges. The host starts by sending the WRITE command byte. The simultaneous response byte may be data from the previous command or it may be a filler byte. The host then sends the byte being written to the register. The simultaneous response byte is 0xFB. Multiple writes must be sent as consecutive single writes.

**Example:** Write the value 0x03 to the Control register (0x04):

*A Write is formatted as 100xxxxx (0x8X). When OR'ed with the register address, the value is 0x84*

$$0x80 \mid 0x04 = 0x84$$

*Send 0x84 followed by 0x03.*

Table 7

Byte	Command (MOSI)	Response (MISO)
1	0x84	Note 1
2	0x03	0xFB

Note 1: The first response byte may be a filler byte or it may be the response to a previous command.

**NOTE: Many registers contain reserved bits that are not used for this application. Changing any reserved bits could result in unexpected behavior. When writing to a register, always use a read-modify-write algorithm that preserves the values of all reserved bits.**

## 4. Touch Detection

### 4.1. Sample Rate

The Crystal Touch: Single Touch controller samples the touch panel 100 times per second, or every 10 ms. If the Data Ready bit is cleared between scans and there is new data to report, the touch data registers will be updated every 10 ms. If the Data Ready bit is not cleared, the new data is discarded.

### 4.2. Z Level

The Z level is a measure of how much the capacitive field has changed. When no finger is near the touch panel, the Z level will be at or near 0. As a finger approaches the top of the touch panel, the Z level will begin to increase. The Z level will continue to increase as more of the surface area of the finger comes into contact with the top surface of the touch panel.

Position data is generated for any Z level greater than 0. Therefore, it is up to the software driver to determine the minimum Z levels that constitute a touch and a release. For more information see "Appendix A – Software Driver Sample".

### 4.3. Z Idle

Once the finger is removed, the Crystal Touch: Single Touch controller can be configured to either immediately cease sending position data, or it can send a specified number of empty packets. This feature is referred to as Z Idle. If configured to send empty packets, the Data Ready signal will be asserted every 10 ms until all the empty packets have been generated.

### 4.4. Capacitor Charge Time

By default, the sense capacitors in the touch panel are enabled for 500 ns. This default value works correctly for all touch panels except the 7.0" diagonal touch panel. For 7.0" touch panels, the capacitor charge time must be changed to 1000 ns.

Changing the capacitor charge time involves reading and writing several two-byte extended register addresses. Reading and writing an extended address is accomplished by writing the two bytes of the address to a pair of standard registers, then writing a command to a third register. Each extended memory access takes several milliseconds to process. Note that the POST bit in the Status register needs to be cleared after each extended memory access.

To change the capacitor charge time to 1000 ns, write 0x08 to extended memory address 0x0EE. This can be accomplished by transmitting the following sequence of bytes to the Crystal Touch: Single Touch controller.



Table 8

MOSI	Description	MISO	Description
0x83	Write to register 0x03	0xFB	Filler byte
0x08	Disable finger tracking	0xFB	Filler byte
<b>Wait 10 ms</b>			
0x9C	Write to register 0x1C	0xFB	Filler byte
0x00	Upper address byte	0xFB	Filler byte
0x9D	Write to register 0x1D	0xFB	Filler byte
0xEE	Lower address byte	0xFB	Filler byte
0x9E	Write to register 0x1E	0xFB	Filler byte
0x01	Read command	0xFB	Filler byte
0xBE	Read from register 0x1E	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler Byte	0x00	Contents of register 0x1E
<b>Repeat reading register 0x1E until it contains 0x00</b>			
0xBB	Read from register 0x1B	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	Value	Contents of register 0x0EE
0x9B	Write to register 0x1B	0xFB	Filler byte
(Value & 0xF0)   0x08	Set charge time to 1000 ns	0xFB	Filler byte
0x9E	Write to register 0x1E	0xFB	Filler byte
0x02	Write command	0xFB	Filler byte
0xBE	Read from register 0x1E	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler Byte	0x00	Contents of register 0x1E
<b>Repeat reading register 0x1E until it contains 0x00</b>			
0x82	Write Status Register	0xFB	Filler byte
0x00	Clear Status register	0xFB	Filler byte
0x83	Write to register 0x03	0xFB	Filler byte
0x00	Enable finger tracking	0xFB	Filler byte
0x87	Write to Calibration	0xFB	Filler byte
0x01	Start calibration	0xFB	Filler byte
<b>Wait 100 ms</b>			
0xA2	Read register 0x02	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	0xFB	Filler byte
0xFB	Filler byte	Value	Status
<b>Continue reading register 0x02 until it is non-zero</b>			
0x82	Write to register 0x02	0xFB	Filler byte
0x00	Clear status	0xFB	Filler byte

## 5. Register Descriptions

### 5.1. Status (0x02)

Table 9

Bit	7	6	5	4	3	2	1	0
Name					POST	DR		
Read/Write					R/W	R/W		
Initial Value					0	0		

- **Bit 3 – POST: Power On Self Test**

The POST bit indicates successful completion of the power on self test (POST) following a reset. It is cleared by writing to the register. The POST bit should only be polled if the host cannot access the hardware Data Ready signal (see “8. Data Ready”).

- **Bit 2 – DR: Data Ready**

The DR bit indicates when new position data is ready. As long as no finger is detected, this bit stays low. When a finger is detected on the surface of the touch panel, this bit is set high every 10 ms as long as the finger is present. It is cleared by writing to the register. The DR bit should only be polled if the host cannot access the hardware Data Ready signal (see “8. Data Ready”).

**NOTE: When clearing the Status register, a read-modify-write algorithm is not required. Simply write 0x00 to the register to clear it.**

### 5.2. Power and Reset (0x03)

Table 10

Bit	7	6	5	4	3	2	1	0
Name						SLEEP	SD	RESET
Read/Write						W	W/R	W
Initial Value						0	0	0

- **Bit 2 – SLEEP: Sleep Mode**

The SLEEP bit is used to enter Sleep mode. When set to 1, the Crystal Touch: Single Touch controller will enter Sleep mode five seconds after the last touch is sensed. Power consumption in Sleep mode is approximately 40  $\mu$ A. The controller will check for a touch every 300 ms. If a touch is detected or an SPI message is received, it will switch back to Active mode. The controller will continue to enter Sleep mode following a touch or an SPI message as long as this bit is set to 1.

- **Bit 1 – SD: Shutdown**

The SD bit is used to enter Shutdown mode. When set to 1, the controller enters Shutdown mode immediately. Power consumption in Shutdown mode is approximately 1  $\mu$ A. The controller will not respond to touches in Shutdown mode.

If a communications message is sent, the controller will switch to Active mode to process the message and then return to Shutdown mode.

- **Bit 0 – RESET: Reset**

Write a 1 to this bit to force a reset of the controller. Follow the standard Initialization process following a reset.

**NOTE: Reset is not supported on the 10.1" touch panel.**

### 5.3. Control (0x04)

Table 11

Bit	7	6	5	4	3	2	1	0
Name	YINV	XINV					A/R	EN
Read/Write	R/W	R/W					R/W	R/W
Initial Value	0	0					0	0

- **Bit 7 – YINV: Y Axis Invert**

Writing a 0 to this bit results in standard reporting for the Y position data (0 to 1535). Writing a 1 to this bit inverts the Y position data (1535 to 0). The correct setting for this bit depends on the orientation of the Crystal Touch: Single Touch panel.

- **Bit 6 – XINV: X Axis Invert**

Writing a 0 to this bit results in standard reporting for the X position data (0 to 2047). Writing a 1 to this bit inverts the X position data (2047 to 0). The correct setting for this bit depends on the orientation of the Crystal Touch: Single Touch panel.

- **Bit 1 – A/R: Absolute / Relative Mode**

Writing a 0 to this bit puts the Crystal Touch: Single Touch controller in Relative (mouse) mode. Writing a 1 to this bit puts the controller in Absolute mode. Ocular recommends always using Absolute mode.

- **Bit 0 – EN: Enable**

Writing a 0 to this bit disables the controller. Writing a one to this bit enables the controller.

### 5.4. Z Idle (0x0A)

Table 12

Bit	7	6	5	4	3	2	1	0
Name	ZI							
Read/Write	R/W							
Initial Value	0x1E (30)							

- **Bits 7 to 0 – ZI: Z Idle**

The number in this register defines how many times the Crystal Touch: Single Touch controller will present empty position data to the host after the finger has been lifted from the touch panel. An empty position is defined as both the X and Y positions set to 0. Because new packets are ready every 10 ms, tap times can be calculated easily by counting the number of empty packets received following a touch.

When set to 0, this register prevents any empty packets from being sent. In this case, the position registers will contain the last sensed location until a new finger presence is detected.

## 5.5. Calibrate (0x07)

Table 13

Bit	7	6	5	4	3	2	1	0
Name								CAL
Read/Write								R/W
Initial Value								0

- **Bit 0 – CAL: Calibrate**

Writing a 1 to this bit forces a self-calibration cycle. Calibration takes approximately 100 ms. When calibration is complete, the POST bit in the Status register will be set. The Status register must be cleared to continue normal operation.

## 5.6. Touch Position Data (0x14, 0x15, and 0x16)

Table 14

Bit	7	6	5	4	3	2	1	0
Name	XDLB							
Read/Write	R							
Initial Value	0xFF							

- **Address 0x14::Bits 7 to 0 – XDLB: X Position Low Byte**

The lower eight bits of the twelve bit X position data.

Table 15

Bit	7	6	5	4	3	2	1	0
Name	YDLB							
Read/Write	R							
Initial Value	0xFF							

- **Address 0x15::Bits 7 to 0 – YDLB: Y Position Low Byte**

The lower eight bits of the twelve bit Y position data.

Table 16

Bit	7	6	5	4	3	2	1	0
Name	YPHN				XPHN			
Read/Write	R				R			
Initial Value	0xFF				0xFF			

- **Address 0x16::Bits 7 to 4 – YPHN: Y Position High Nibble**

The upper four bits of the twelve bit Y position data.

- **Address 0x16::Bits 3 to 0 – XPHN: X Position High Nibble**

The upper four bits of the twelve bit X position data.

For all Crystal Touch: Single Touch panels, the X coordinate ranges from 0 to 2047 and the Y coordinate ranges from 0 to 1535. It is the host's responsibility to scale these values to the display resolution being used.

## 5.7. Z Level (0x17)

Table 17

Bit	7	6	5	4	3	2	1	0
Name			ZL					
Read/Write			R					
Initial Value			00					

- **Bits 5 to 0 – ZL: Z Level**

In addition to X and Y position data, the Crystal Touch: Single Touch controller measures finger presence on the touch panel surface. The signal strength of the finger is called Z level. A smaller finger results in a lower Z level and a larger finger gives a larger Z level. See section 4.2 Z Level.

Z level values range from 0 to 63.

## 6. Initialization

The Crystal Touch: Single Touch controller performs a power on self test (POST) and a self calibration following a reset. If the POST completes successfully, the POST bit in the Status register (0x02) is set and the Data Ready signal is set high.

Once the POST has completed, the controller needs to be set to absolute mode and enabled. This is done by writing the value 0x03 to the Control register (0x04). You may also need to invert the X or Y axes which are controlled by the highest two bits in this register (see "5.3. Control (0x04)").

Next, clear the POST bit, Data Ready bit, and Data ready signal by writing the value 0x00 to the Status register (0x02). If using a 7.0" panel, next change the capacitor charge time to 1000 ns (see "4.4. Capacitor Charge Time").

## 7. Coordinate Scaling

Although the Crystal Touch: Single Touch controller is theoretically capable of returning coordinates between (0, 0) and (2047, 1535), due to the geometric construction of the touch panels, the actual reported coordinates in the corners of the touch panel are typically closer to (64, 64) and (1983, 1471).

This origin shift must be accounted for when scaling the panel's coordinates to the display's coordinates. First, shift the reported coordinates to an origin of (0, 0) by subtracting 64 from each value. Then divide by the maximum coordinate in X and Y. Keep in mind that the maximum coordinates should also be shifted by 64, meaning that the maximum X value is  $1983 - 64 = 1919$ ; the maximum Y value is  $1471 - 64 = 1407$ .

The following code example scales the panel coordinates to a 16:9 display with a width (X) of 533 and a height (Y) of 300:

```
// check for underflow
if (PanelX < 64) PanelX = 0;
else PanelX -= 64;

if (PanelY < 64) PanelY = 0;
else PanelY -= 64;

// scale coordinates
DisplayX = (PanelX * 533) / 1919; // 1919 = 2047 - 128
DisplayY = (PanelY * 300) / 1407; // 1407 = 1535 - 128

// check for overflow
if (DisplayX > 532) DisplayX = 532;
if (DisplayY > 299) DisplayY = 299;
```

Note the underflow check on the panel coordinates and the overflow check on the display coordinates. This is required because the panel might report coordinates slightly less than (64, 64) or slightly greater than (1983, 1471).

## 8. Data Ready

In addition to the Data Ready software flag in the Status register (0x02), there is also a Data Ready hardware signal (DR). The Crystal Touch: Single Touch controller drives this signal high when new position data is available. New position data may be available as often as every 10 ms.

Once the new data has been read, write 0x00 to the Status register (0x02) to clear the DR signal.

This signal also indicates that a Power On Self Test (POST) has completed successfully following a reset. See "6. Initialization".

## Appendix A – Software Driver Sample

The following sample code demonstrates how to use the Z level to determine when a touch and release event occurs.

```
bool TouchPanelDataIsReady(void);

struct POSITION_DATA
{
    uint16 X, Y;
    uint8 ZLevel;
};
void GetFingerPosition(struct POSITION_DATA *Position);

/*
    These are typical numbers. You made need to
    tune them to your application and enclosure.
*/
#define MIN_Z_LEVEL_FOR_TOUCH      25
#define Z_LEVEL_RELEASE_THRESHOLD  20

static bool FingerIsPresent = FALSE;

if (TouchPanelDataIsReady())
{
    struct POSITION_DATA FingerPosition;
    GetFingerPosition(&FingerPosition);

    if (FingerIsPresent)
    {
        if (FingerPosition.ZLevel < Z_LEVEL_RELEASE_THRESHOLD)
        {
            FingerIsPresent = FALSE;

            //////////////////////////////////////
            // post finger up event to OS //
            //////////////////////////////////////
        } /* if-then */
        else
        {
            //////////////////////////////////////
            // post finger position data to OS //
            //////////////////////////////////////
        } /* if-else */
    } /* if-then */
    else
    {
        if (PositionData.ZLevel > MIN_Z_LEVEL_FOR_TOUCH)
        {
            FingerIsPresent = TRUE;

            //////////////////////////////////////
            // post finger down event and data to OS //
            //////////////////////////////////////
        } /* if-then */
    } /* if-else */
} /* if-then */
```